Pergamon

0098-3004(95)00020-8

Computers & Geosciences Vol. 21, No. 7, pp. 877–894, 1995

0098-3004/95 $9.50 + 0.00

ESTIMATING PROBABILITY SURFACES FOR GEOGRAPHICAL POINT DATA: AN ADAPTIVE KERNEL ALGORITHM

CHRIS BRUNSDON

Department of Town and Country Planning, University of Newcastle upon Tyne, NE1 7RU, U.K.
(*e-mail*: Chris.Brunsdon@newcastle.ac.uk)

(Received 24 August 1994; revised 20 November 1994)

Abstract—The statistical analysis of spatially referenced information has been acknowledged as an important component of geographical data processing. With the arrival of GIS there has been a need to devise statistical methods that are compatible with, and relevant to, GIS-based methodologies. Here an algorithm is presented which estimates a "risk surface" from a set of point-referenced events. Such a surface may be viewed as an object embedded in three-dimensional space, or as a contour map. In addition to this view, it is possible to incorporate these surfaces into a broader based GIS framework, allowing the mapping of these patterns in conjunction with other data, overlay analysis, and spatial query. The technique is adaptive, in the sense that parameters which control the surface estimation are adjusted over geographic space, allowing for local variations in point pattern characteristics. The paper is concluded with an example based on probabilistic mapping using data taken from Californian Redwood seedling data.

Key Words: Density estimation, Spatial analysis, Kernel, GIS.

INTRODUCTION

Recent years have seen great advances in the capabilities of Geographical Information Systems (GIS). These software packages provide powerful tools for the manipulation and presentation of spatial data, and recently have incorporated features allowing some forms of spatial analysis. For example, many features of the GRID module in the ARC/INFO GIS allow the analysis of networks and transport costs to be carried out within the package. These techniques allow, for example, the computation of cheapest routes from one location to another, or indices of accessibility for a study region. Although this is useful, most of these analytical techniques require the provision of exhaustive and accurate geographical data over the region under study. In many other situations complete data may not be available, so that sample-based analyses must be employed. In these situations some analysis of levels of uncertainty may be required to interpret the spatial patterns observed. However, few GIS provide facilities for the statistical analysis of spatial data or for the analysis of probabilistic data. In an era where "... the GIS revolution is generating vast amounts of spatial information and geography-relevant data" Openshaw (1994) it may seem that there is little demand for techniques of this sort. However there are many situations where the data obtained are only a subset of a population for some geographical phenomenon, possibly because not all incidences of that phenomenon can be detected, or because of the prohibitive cost of data collection. For example, it may not be possible to obtain details of all household burglaries occurring in a given area, because some may not be reported to the police, or the data may have been collected by an insurance company who would only record burglary claims for their own policy holders.

Many researchers in the GIS field have commented on the lack of probabilistic analytical statistical functionality within GIS packages, most notably Fotheringham and Rogerson (1993) and Openshaw (1990), and have called for the provision of software to meet this shortcoming. The opinion also has been expressed that any statistical analysis techniques that are used in conjunction with a GIS should retain the general character of both the GIS and statistical disciplines. Thus, although considering data in a probabilistic sense, the analysis also should provide graphical, man-based output. Some statistical techniques already in existence may be adapted easily to this GIS framework, but others less so. The possibility that new statistical techniques may need development also must be considered.

A particularly important factor when considering any statistical technique for linkage are the assumptions that are made when it is applied. For example, statistical methods which assume a

Gaussian distribution in their study data at least will have to be checked for robustness in situations where this assumption is violated. More fundamentally, techniques which "assume away" the geography of the situation should be avoided—in particular, the supposition that observations related to a set of geographical zones are distributed independently is unlikely to be true in many geographical contexts.

Another shift in character from traditional statistical techniques to a GIS approach is a greater emphasis on location itself. In the former situation, regression models, principal component analyses, and other related techniques are applied to the attributes of points or zones in space, but far less the location itself is treated as a random variable. Although the work of Anselin (1988) and others on spatial regression, which rejects the independence assumption, assumes that the zonal system is fixed and without error. However, it is perhaps in the analysis of location itself that statistical techniques have most to offer the GIS community. Much of the output of GIS-based analysis is in cartographic form, answering questions as to where certain conditions are met, or where certain phenomena occur; the ability to estimate population spatial patterns from a sample, or test hypotheses based on locational characteristics would fit into a wider GIS framework, and could perhaps be offered by some well-selected statistical techniques.

In this paper, one such method is proposed. The kernel-based approach to probability density function estimation (Silverman, 1986; Diggle, 1985, 1990) provides a tool for determining the shape of a probability density distribution given a set of points that have been drawn from that distribution. The method can be applied to points in any dimension of Euclidean space. Most importantly for geographical analysis, it may be applied specifically to points in two-dimensional space. Given a set of locations of some phenomena, such as the locations of household burglaries or road accidents, it is possible to estimate a "risk surface" showing the relative likelihood that further events have of occurring in various parts of the study region. This form of statistical analysis can provide cartographic output, and addresses spatial randomness by treating the location of events as the outcome of a two-dimensional random process. It also may be thought of as estimating a "population" spatial pattern from a sample. For example, by applying the technique to a sample of household burglary locations for a fixed term study period, estimations relating to probability patterns for all household burglaries in an area may be made. Finally, the technique may be thought of as nonparametric, or distribution free, because rather than making any distributional assumptions about the point data patterns, it attempts to estimate these from the data themselves.

STATISTICAL BACKGROUND

Overview of kernel density estimation

Before discussing the computational techniques used to carry out this analysis, some statistical background is presented. As its name suggests, the kernel density estimation technique provides for estimating the probability density function f of a distribution from which a sample $\{x_i\}$ has been observed. This is done by centering a probability density function k (the kernel distribution function) around each of the observed points, and then taking the average value of all of these. k usually is selected to be unimodal and symmetrical. As is demonstrated visually in Figure 1, this average value gives a form of estimate of the probability distribution of the sample points. Mathematically, the estimate may be expressed as

$$f(x) = \frac{1}{nh} \sum_{i=1}^{n} k\left(\frac{x - x_i}{h}\right) \tag{1}$$

where n is the number of observations in the sample. This in itself, when extended to two dimensions, can provide the basis for a cartographic technique. For discussion and an example of this extension, see Langford and Unwin (1994). However, Equation (1) leaves some unresolved problems. The variable h is of key importance here. Usually referred to as the bandwidth or radius of the kernel estimation, this parameter controls the variance of the density function k. Large values of h will cause k to have a large variance, whereas smaller values will have the opposite effect. The implications of this are illustrated in Figures 2 and 3. Clearly, when h has an unacceptably large value (Fig. 2), the estimated distribution f becomes nondescript, and shows no response to changes in density of the observations $\{x_i\}$. In fact, if the value of h is greater than the range of the x_i's, f will take on the shape of k regardless of any patterns

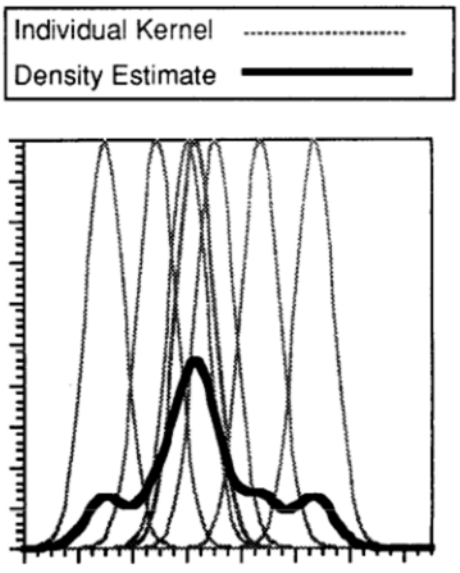

Figure 1. Kernel density estimation method.

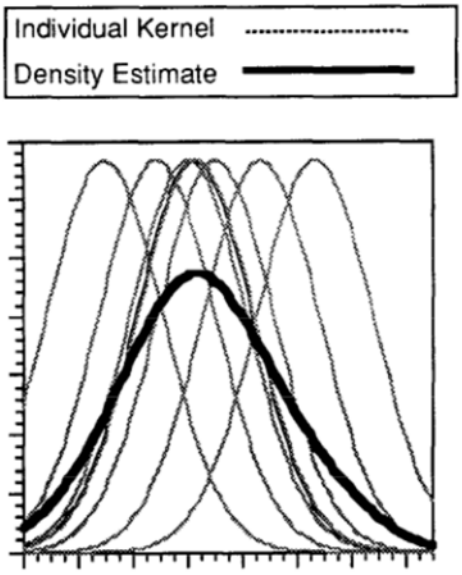

Figure 2. Result of too large bandwidth.

in the data. Figure 3 shows the opposite effect. If h is too small, for example having a value less than the average separation of the x_i's, then f will appear to be a series of spikes centered on the observed data points. The value of h used in Figure 1 gives a more reasonable f, but these three examples suggest that there should be some "optimal" (or at least automatic) way of selecting h.

One such method is that of cross-validation. This draws its inspiration from the idea of maximum likelihood estimation. In maximum likelihood estimation, the probability density function for some variate x is expressed as $f(x; \boldsymbol{a})$ where $\boldsymbol{a}$ is a vector of parameters for some distribution f. The likelihood of a set of observations $\{x_i\}$ is the product of these probability densities for each x_i:

$$l(\{x_i\}; a) = \prod_{i=1}^{n} f(x_i; a). \tag{2}$$

One method for estimating $\boldsymbol{a}$ is to select a value which maximizes the product. Although in this situation f is not of a known form, it may be thought of as being dependent on the value of h. It therefore is possible to write a likelihood function $l(\{x_i\}; h)$ which could be maximized in terms of h. There is, however, one difficulty. Noting the form of Equation (1), consider the value of $f(x_i)$, that is the estimate of the probability density function at any of the sample points. This is an average of kernels, and the sum may be split into the kernel around x_i, and the remaining kernels:

$$f(x_i) = \frac{1}{nk} k(0) + \sum_{j=1, j \neq i}^{n} \frac{1}{nh} k\left(\frac{x_i - x_j}{h}\right). \tag{3}$$

The first term is a function of zero because the numerator to the argument of k is proportional to $x_i - x_i$. Note that as the value of h tends to zero, the value of $f(x_i)$ tends to infinity. Thus, applying the maximum likelihood method directly would suggest that zero is an optimal value of h. Figure 3 suggests that this would not be appropriate, as it yields a series of Dirac spikes positioned on the observation points. (Dirac spikes may be defined informally as improper functions whose value is "infinite" at a single point on the real line, and zero elsewhere.) A more satisfactory result may be obtained if the first term in Equation (3) were dropped from the estimate of $f(x_i)$. This is not an unreasonable action. It effectively defines a new estimate of f, denoted here as $f^{[i]}$ as

$$f^{[i]}(x) = \frac{1}{nh} \sum_{j=1, j \neq i}^{n} k\left(\frac{x - x_j}{h}\right) \tag{4}$$

which is an estimate of the probability density function based on all values in the sample except the ith. If the likelihood function is modified slightly to

$$l^*(\{x_i\}; h) = \prod_{i=1}^{n} f^{[i]}(x_i; h). \tag{5}$$

This can be thought of as a likelihood estimate using the kernel technique to estimate the probability density at each x_i given all of the remaining observed values. The avoidance of each individual data point in the estimation of density at that point itself removes the problematic term in Equation (3), and allows more suitable estimates of h to be achieved. Because the technique involves estimating the value of the probability density for each x_i in terms of the rest of the data set, the term cross validation is applied.

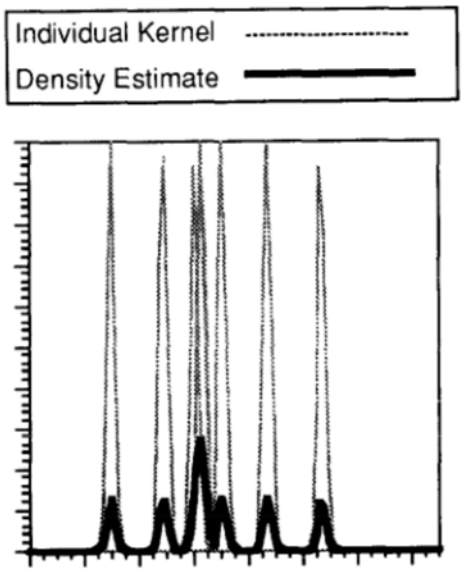

Figure 3. Result of too small bandwidth.

Making kernel estimates adaptive

The technique may be enhanced further by allowing the value of h to range between different regions

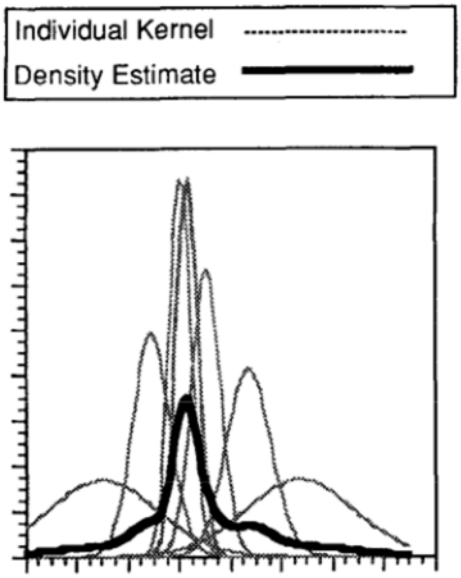

Figure 4. Adaptive kernel density estimate.

of the sample space. In this situation, the density estimation function becomes

$$f(x) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{h_i}k\left(\frac{x - x_i}{h_i}\right) \tag{6}$$

where h_i varies with x_i. This is useful when the distributional density differs notably within the sample region. If points are close together, it may be more appropriate to use a smaller value of h, as this will reduce the effects of over-smoothing and allow the estimated density to be more responsive to subtler details of variation within the sample (see Figs 4 and 5 for an example). This is relevant particularly to human geographical data—most events to be analyzed will occur in the most densely populated areas, and it is in these areas that identification of variations in probability density will be of most interest. Thus,

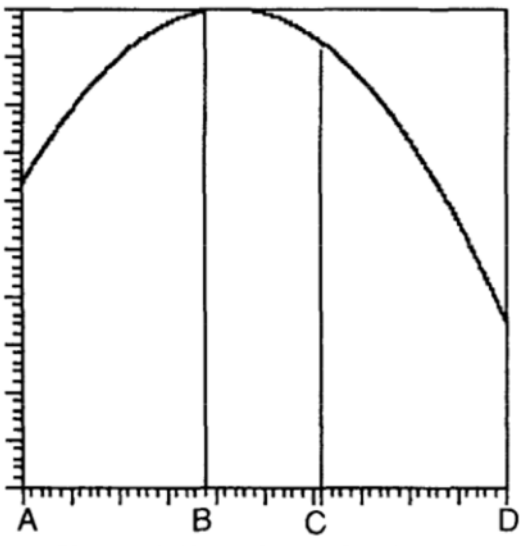

Figure 5. Golden section search optimation.

some system for reducing the h_i values in high-density regions, and increasing them in low-density regions should be sought. A plausible way of achieving this may be set out in two steps:

- Compute a fixed-h estimate of density for each x_i
- Use these estimates to obtain an h_i at each of these points

One possible method which could be used for the second step is to compute h_i as $l_i h$ where

$$l_i = \left(\frac{f^*(x_i)}{h}\right)^{\alpha} \tag{7}$$

and f^* is a fixed-h density estimate. This follows from Silverman (1986), who suggests that α should take a negative value. The selection of the initial value for h has been determined not to affect greatly the final result. Again, an appropriate selection for α may be determined using cross-validation techniques. This is investigated in greater detail later in the paper.

The remaining problem is to implement these techniques in two-dimensional space. All of the formulae so far have considered estimating a scalar density function. Fortunately, most of the adaptation of these for the two-dimensional situation involves replacing the scalar x with a two-dimensional vector $\mathbf{x}$. The only additional consideration is the kernel density function. In the two-dimensional situation, the kernel function has a circle of influence around the location of each observed point which decays with distance. In mathematical terms, this may be expressed as

$$f(\mathbf{x}) = \frac{1}{nh}\sum_{i=1}^{n}k\left(\frac{|\mathbf{x} - \mathbf{x}_i|}{h}\right) \tag{8}$$

the only changes are that the scalar difference of the sample point and evaluation point is replaced by a vector distance and h is replaced by h^2 in order to normalize a two-dimensional density function k. Substitutions of this form may be made to Equations (1)–(7) to estimate the corresponding two-dimensional densities.

IMPLEMENTATION ISSUES

Working with SAS

In this paper, two methods of computation are proposed. Firstly, the spatial data may be analyzed using a statistical package, SAS (The SAS Institute, 1994). Secondly a more conventional programming language may be used. Here, sample code is supplied in FORTRAN 77. The justification for the first approach is that SAS provides a powerful language for handling spatial data—the Interactive Matrix Language (or IML), and when the analysis has been completed the results may be fed seamlessly into graphical or mapping facilities within SAS.

However, although the use of powerful interpreted languages may allow concise solutions to computational problems of this type, compiled languages

will become necessary when working with larger data sets. For this reason, the algorithm has been coded in FORTRAN 77. An added advantage is that the stand-alone program may be integrated into the ARC/INFO GIS environment, allowing the results of the distribution estimation to be manipulated within a GIS. Although this is not as simple to use as the SAS macro, larger data sets may be analyzed, and integrated with other geographical information. Working in FORTRAN 77 has the added advantage that spatial data may be manipulated as complex numbers; this can lead to a simplicity of expression for algorithms which are applied to spatial data.

Computational methods

A model for computation which applies well to kernel-based density estimation is the tableau. In this approach, intermediate results are arranged in a rectangular table, similar to a spreadsheet (although not necessarily visible to the user), and the final results may be expressed in terms of mathematical functions applied to the elements of the table, or on a row-by-row or column-by-column basis. In this situation, each row in the tableau is associated with the location of one of the observations in the sample data set and each column with the location of an estimation point. In the first instance, these estimation points can be spaced on a regular lattice within a rectangular zone over which the densities are to be investigated. A distance table is computed between row data points and column evaluation points. This is equivalent to the $|\mathbf{x}-\mathbf{x}_i|$ term in Equation (8). Next, for each cell in the table, the function k is applied, using a prespecified value for h. Thus, for each evaluation point column, the ith row of the tableau contains the ith term in Equation (1). Averaging down the columns therefore gives the kernel density estimate for each estimation point.

This can be implemented relatively easily in IML (see Listing 1 in the Appendix). The availability of the Kronecker product operator is useful particularly in this calculation. The Kronecker product of two matrices is obtained by multiplying the second matrix by each element of the first in turn, with these new matrices laminated in the same relative positions as the elements of the first matrix. Algebraically:

$$\mathbf{a}\otimes\mathbf{b}=\begin{bmatrix} a_{11}\mathbf{b} & a_{12}\mathbf{b} & \cdot & \cdot & a_{1c}\mathbf{b} \\ a_{21}\mathbf{b} & \cdot & & & \\ \cdot & & \cdot & & \\ \cdot & & & \cdot & \\ a_{r1}\mathbf{b} & & & & a_{rc}\mathbf{b} \end{bmatrix} \tag{9}$$

where $\otimes$ is the symbol for the Kronecker product operation, and the matrix $\mathbf{a}$ has r rows and c columns. Note that each item in the square brackets on the right-hand side are themselves matrices not scalar elements of a matrix.

This operator comes into its own when either $\mathbf{a}$ or $\mathbf{b}$ is a row or column vector of ones. In this situation, the other matrix will be repeated n times either vertically or horizontally in the Kronecker product. Combining one matrix of vertical repetitions of the evaluation points with another of horizontal repetition of the sample points on an element-by-element basis allows each sample point/evaluation point interaction to be considered. For example, subtracting the first matrix from the second gives a difference tableau. In IML this may be programmed as:

```
diff = j(1, r, 1)@a - j(1, 1, c)@b;
```

where the @ symbol denotes the Kronecker product, and the $j(x, y, z)$ function provides a uniformly x-valued matrix of y rows and z columns. If the kernel function were Gaussian, the next step is to apply this element-by-element to the variable diff using a prespecified value for h.

```
diff = exp(-diff ## 2/h)/(h * sqrtpi);
```

here, the double hash operator is an element-wise exponentiation operator, and sqrtpi is a normalizing constant. The more conventional double asterisk is reserved for matrix exponentiation. It remains only to average the values down the columns to obtain the density estimates at the specified points. For such calculations, there are a number of reduction operators in IML; that is operators that reduce the dimension of a matrix by applying some row or column-wise function and returning a scalar value. One of these is the mean operator, which may be used as shown here:

```
estimate = diff[, #];
```

the hash in the column index position denotes that the mean function is to be applied to each column. Thus, using IML it is possible to carry out a simple kernel density estimate in just three lines of code. For brevity the one-dimensional situation is shown, but extension into two dimensions is relatively straightforward (Listing 1 in the Appendix).

It is a similar problem to code the cross-validation estimation technique in IML. Essential to this is the evaluation of the $f^{[i]}$ values at each point x_i. In this situation the tableau is square, because the values at which the function is to be evaluated are the same as those observed in the sample. Because the density estimates at each sample point miss out the density contribution from that point itself, the leading diagonal must be removed from the tableau. A simple way of doing this is to multiply each element in the leading diagonal by zero before averaging the columns:

```
diff = diff # [j(1, n, n) - i(n)];
```

The single hash denotes elemental multiplication, and the i function gives an n-dimensional identity matrix. After the averages have been computed, the cross-validation likelihood may be estimated, as in

Equation (7). Because multiplying several probability densities together is likely to result in extremely small numbers, the sum of their logarithms is actually computed. This is relatively straightforward in IML, using the + reduction operator:

$$l = \ln(\text{estimate});$$

$$\text{lhood} = l[+];$$

Clearly, this likelihood can be viewed as a function of the kernel bandwidth, h. Maximizing this quantity is equivalent to maximizing the likelihood product in Equation (7), and so by maximizing l in terms of h, an "optimal" bandwidth may be selected.

The next matter for consideration is the method used to determine the value of h maximizing l. For some kernels, it may be possible (although complicated) to use Newton's method (see for example, Hosking, Joyce, and Turner, 1981), to locate a zero of the first derivative of l with respect to h. A more general, numerical-based solution is to use a golden section search (Baxter, 1976; Greig, 1980). In this method, if a range of h is given which is known to contain a single maximum, this range is subdivided into three sections, delimited by the two end points and two intermediate points B and C in positions $1/(1+\sqrt{5})$ of the length AD from A and D respectively. Depending on whether the value of the function at B exceeds that at C, the search region for the maximum is shrunk either to AC or BD. If B has the larger value, as in Figure 5, then AC is selected, otherwise BD is selected. As Greig (1980) demonstrated the positions of B and C are selected to minimize the maximum potential width of the new interval. Clearly, each time this operation is repeated, the size of the interval containing the maximum value is reduced by a factor of $2/(1+\sqrt{5})$, so that the maximum can be located to any specified tolerance if the golden section subdivision is repeated a sufficient number of times.

To implement this here, the cross-validation likelihood needs to be computed for h values selected according to the golden section search methodology. This is achieved by encapsulating the likelihood computation in an IML subroutine, taking h as a parameter, and returning the likelihood value, and then calling this subroutine in a loop until the upper and lower limits for the "optimal" h-value are sufficiently close together.

Having selected an optimal value of h, one final task remains; this is to compute the kernel density estimates at a set of evaluation points. This requires just one more pass of the kernel tableau procedure, this time evaluated conventionally (not excluding any observations as with the cross-validation), as in the earlier simple kernel routine. The entire process is carried out by the code in Listing 2 (see the Appendix).

The final problem is to compute the adaptive kernel estimates for a given set of evaluation points. Again, this will require the tableau methodology to be employed. However, in this situation h is not thought of as a scalar, because its value is different for each point in the sample of points. Thus, in the tableau format, the computation of the kernel densities is not repeated for the entire table; a different h-value is required for each row. If the h-values are stored in a column vector termed hvec then the IML code to compute a Gaussian kernel is

$$hk = \text{hvec}@j\,(1, 1, c);$$

$$\text{diff} = \exp(-\text{diff} \# \ \# \ 2/h)/(h * \text{sqrtpi});$$

The first line duplicates the column vector along each of the columns, and the second computes the kernel densities on an element-by-element basis. Again, note that the Kronecker product is used in this computation. To obtain the density estimates, column averages are taken once again. This entire process of computation is employed in Listing 3 (see Appendix) a SAS IML program to obtain adaptive kernel estimates at a set of evaluation points, given a point data sample.

Working with a traditional compiled language

In Listings 1–3 (see the Appendix), techniques in SAS/IML are given which obtain two-dimensional kernel estimates. As discussed earlier, these may be integrated into the SAS environment, which provides statistical analysis, graphical, cartographic, and data manipulation functionality. However, as this is an interpreted language, there may be times when a data set is large enough to make computation in SAS/IML prohibitively slow. In such situations, a compiled computer language should be used. To demonstrate how density estimation techniques may be used from this viewpoint, implementation of the given methods in FORTRAN 77 now will be considered. The particular flavor of FORTRAN 77 used here is Sun FORTRAN, with VAX extensions enabled, running on a Sun Sparcstation 10.

When working with traditional compiled languages of this sort, a helpful way to implement a suite of techniques such as these is to code them as a subroutine library. Then, certain "driver" programs are used to call these routines, and output the results. The technique is helpful particularly here, because factors that are liable to differ from problem to problem, such as the format of the input data, or the graphical environment used to present the output, may be dealt with in the calling program. However, the invariant part of the problem, the density estimation algorithm itself, may remain encapsulated in the library.

The subroutine library is given in Listing 4 (see Appendix). The computation techniques are equivalent to those given in the section on Statistical Background making use of the tableau concept. However, as FORTRAN 77 does not offer the luxuries of array handling provided in SAS/IML, much more use of explicit do-loops occurs. However, as mentioned

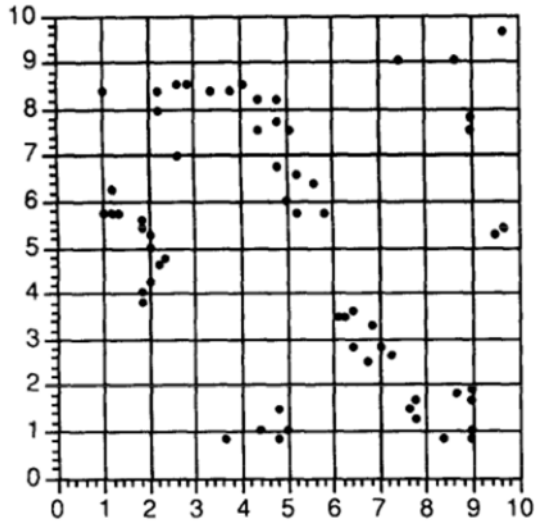

Figure 6. Location Californian Redwoods (Diggle, 1984).

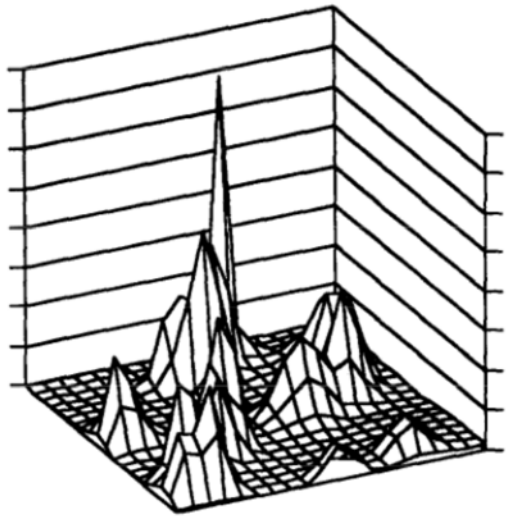

Figure 8. Using three-dimensional surfaces to show probability densities.

earlier, the use of complex variables does help to simplify the implementation of these algorithms in two-dimensional space.

A CASE STUDY: THE CALIFORNIA REDWOOD SEEDLING LOCATION DATA

The data set

Similar to most computational techniques, kernel density estimation is best illustrated with a practical example. In this case, the data relating to the locations of Californian Redwood seedlings will be used to demonstrate the method put into practice. This data set, which was extracted by Ripley from a larger data set in Strauss (1975), relates to the locations of Redwood seedlings in a 1 km square sample quadrant. It was used subsequently as an example data set by Diggle in his 1983 book on point pattern analysis. For each tree, its coordinate pair is recorded in terms of northern and eastern displacement from the southwestern corner of the sample zone. The tree locations are shown in Figure 6. Here, the location of the trees will be considered as a random process: in some regions trees are more likely to survive than others. The actual location of the trees therefore can be thought of as a realization of a random spatial process. The task for this case study is to estimate the probability density function associated with this process.

Graphing the analysis results

Graphical output is an important product of this type of analysis—the probability density function may be thought of as a mathematical function over geographical space, and visual appearance perhaps is the most important key to its interpretation.

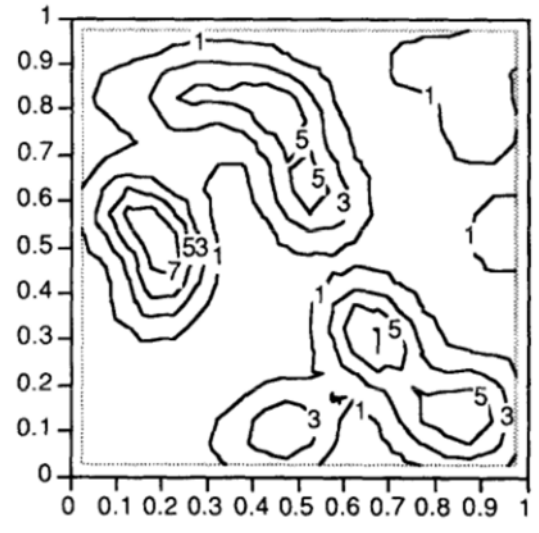

Figure 7. Using contours to show probability density.

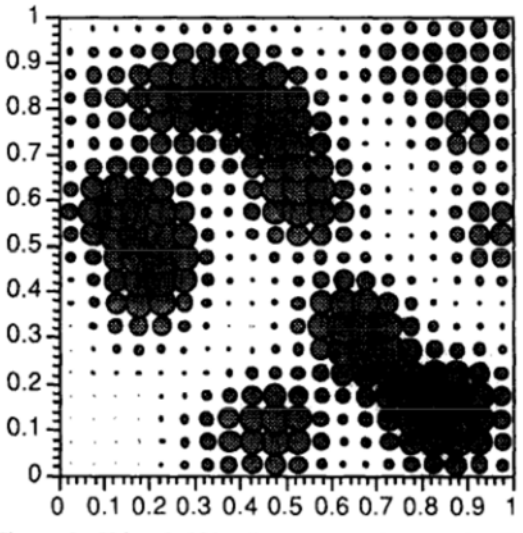

Figure 9. Using bubble diagrams to show probability densities.

However, there are many types of graphical technique used to represent two-dimensional surfaces in three-dimensional space. Because these all have notably different appearances, some thought must be given to the selection of method when presenting results of this sort. Three possible presentation methods are suggested here:

- Contouring (Fig. 7)
- Three-dimensional perspective drawing (Fig. 8)
- Proportional area 'bubbles' on a regular grid (Fig. 9)
- A gray-scale map (Fig. 10)—as used for example in Gatrell (1994).

Although usually the most popular method, the perspective drawing has the disadvantage that when viewed from some directions, part of its features are obscured. This possibly can be overcome in an interactive environment, where the viewer could use controls provided by a computer program to rotate the surface, but this is not the situation with printed matter! Contouring overcomes this problem, but it is not immediately clear which areas on the contour map are peaks and which are troughs. This is resolved in the gray-scale map, but in this situation it is difficult to relate relative levels of probability to the change in gray scale. Although all of the other surface representations are useful in specific applications, it has been decided here to use the "bubble plot" option to illustrate the surfaces computed in this case study.

Fixed radius kernel density estimation

Initially, a simple kernel estimate for the density estimation will be carried out, based on the KERN1 subroutine (in Listing 4 in the Appendix). In the first instance the k value selected will be the mean nearest neighbor distance between the trees. This gives the probability surface shown in Figure 9. However, this may be thought of as an arbitrary choice of k. A more rigorous approach may be to apply the cross-validation likelihood technique. A graph of k vs. cross-validation likelihood is shown in Figure 11. As can be seen, an optimal value for k is around 0.062 km. Using the golden section search technique, a more accurate value of k of 0.0626 km is determined. The surface obtained using this value is shown in Figure 12. Clearly, this value is notably more peaked than the "first guess" method based on the mean nearest neighbor distance.

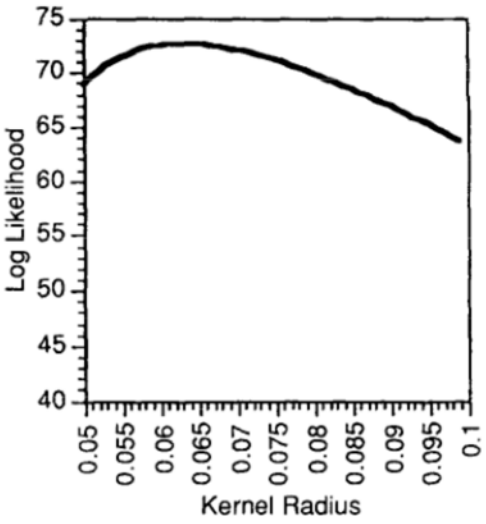

Figure 11. Log cross-validation likelihood vs. kernel radius.

Variable radius kernel density estimation

In this section, density estimation based on the adaptive techniques outlined in the section on making kernel estimates adaptive is carried out. Recall that two parameters have to be specified in this situation—a bandwidth for the initial fixed-radius estimate and then a value for α, the exponent on the

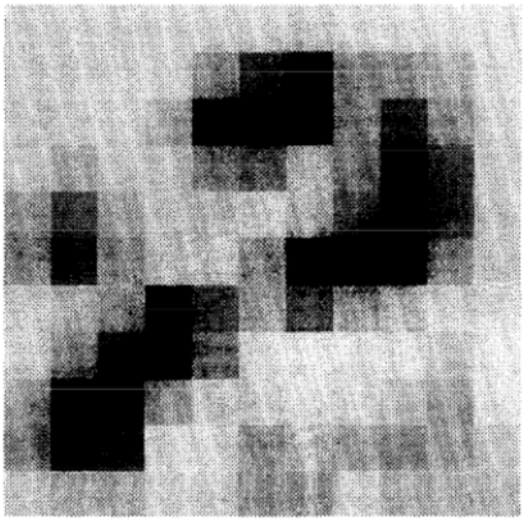

Figure 10. Using gray-scaled pixels to map densities.

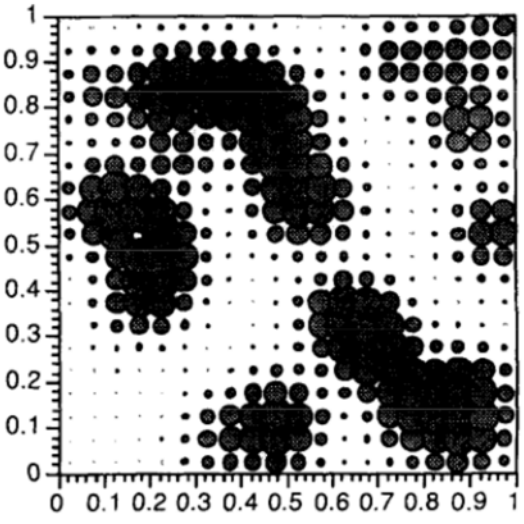

Figure 12. Probability density for cross-validated fixed radius kernel estimate.

adaptive bandwidth adjustment formula. Arbitrary selection for both of these would be difficult to make, so once again the cross-validation likelihood method may be used. Figure 13 shows the value of the likelihood (on a contour graph), for α values between -2 and -1, and initial k-values between 0.04 and 0.06. It may be seen from this graph that the cross-validation likelihood is optimized when k is about 0.05 and α about -1.5.

To obtain more accurate values for these parameters, a more rigorous search routine is employed. In this situation it is the simple stepping search of Hooke and Jeeves (1961). An initial guess is made for optimal values of k and α, and the cross-validation function is evaluated at this point, and at points $\pm h_1$ from k and $\pm h_2$ from α. If the value of the likelihood is higher at one of these sample points, the optimal guess for the parameters will be updated accordingly. If the original guess faired better than the sample points, then each h is halved and the search continues. This is repeated until the h values are considered to be sufficiently small.

This gives a good example of the utility of the subroutine library approach. The general "building blocks" for the adaptive techniques are coded in the KERNLIB library, but the more problem-specific Hooke and Jeeves algorithm is coded in the calling program (Listing 5 in the Appendix). Note that once again the two-dimensional nature of this problem suggests that using variables of type COMPLEX will simplify the coding of the algorithm.

Running this program gives the optimal parameter estimates as $k = 0.0478$ and $\alpha = -1.48$. The convergence of the Hooke and Jeeves search for these values is shown in Figure 14. The density estimate itself is illustrated in Figure 15. The main distinction between the appearance of this estimate and the optimal fixed

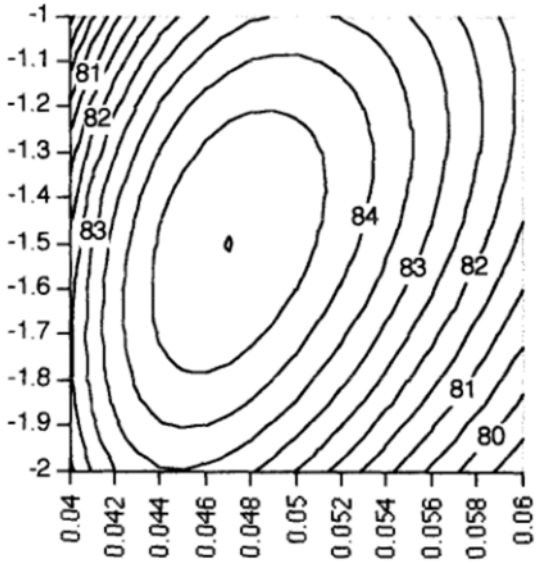

Figure 13. Cross-validation likelihood contours vs. alpha and kernel radius.

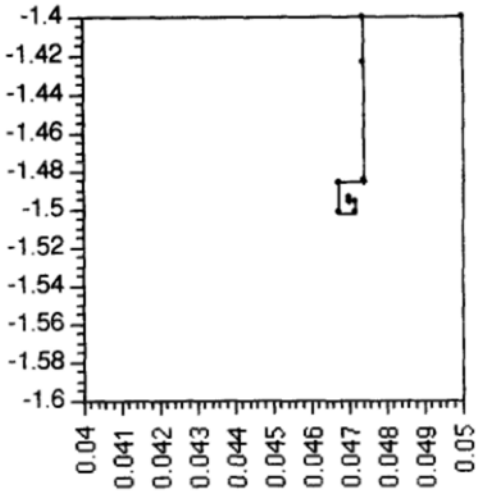

Figure 14. Convergence of Hooke and Jeeves' algorithm (1961).

radius estimate is that there is a greater tendency for sharp edges to occur in this distribution. In areas where a dense cluster of trees is neighbored by barren land, the adaptive method tends to use low bandwidth value, ensuring that the "spillage" of density resulting from the kernels is not great. In the non-adaptive technique, this would not be possible without the penalty of obtaining "spiky" estimates in other less densely clustered areas of trees.

CONCLUSIONS AND DISCUSSION

This paper has provided a method for estimating a spatial probability distribution, given an initial set of points generated from this distribution. Unlike many techniques this also has the ability to alter the parameters of surface estimation in a geographically adaptive way. As stated in the introduction to the paper, there is a need to consider linkages between

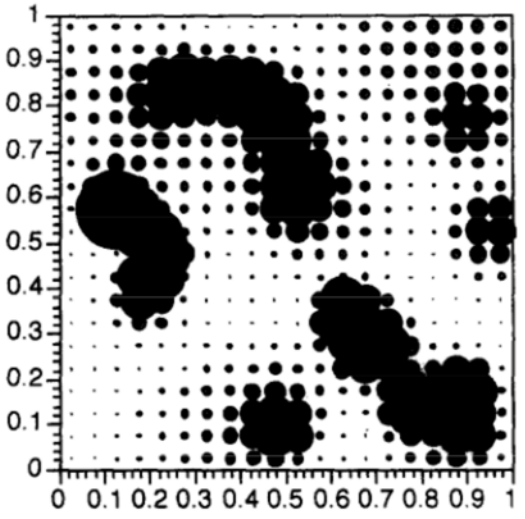

Figure 15. Adaptive kernel density estimate.

statistical methodologies and GIS. It is believed that the technique presented in this paper is one such linkage. Firstly, its adaptive nature reflects the general geographic context that all places differ—allowing at least quantitative representation of "place" as well as "space". Secondly—and perhaps more significantly for the GIS community—it provides a novel form of map layer which may be incorporated with other geographical information.

For example, in the last section a probability surface giving the likelihood of Californian Redwood trees growing in a particular area was estimated. Consider the context of this probability. The event that a tree grows on a particular spot could be dependent on many factors—the spatial distribution of seeds falling in the locality, the condition of the ground on which the seeds fall, the terrain of local land (which governs the amount of sunlight received in different regions), and several others. A research question arising from the case study might be to determine the degree with which each of these factors effects the probability of a tree growing. One thing that all of the factors have in common is that they are all geographical variables, and may be represented in map form. If this information were available in the study quadrant, it could be stored as a series of coverages in a GIS. Combining this information with the output from the adaptive kernel algorithm would allow associations to be explored in a cartographic sense.

The linkage of output from this algorithm to a GIS presents few problems. If a program is produced which produces as output an ASCII file listing probability densities in grid format, it is relatively simple to convert this into the internal format of most GIS packages. For example, in the ARC/INFO package, the command "asciigrid" performs the task. The operation could be automated to a further degree by creating a macro in the ARC/INFO Macro Language (AML) to run the FORTRAN 77 program and feed its output into the conversion routine in a single command. When this task has been completed, the results of the kernel estimation may be combined with any other data stored in the GIS, using various display techniques incorporated in the GIS. Two examples are given here, (Figs. 16 and 17) showing ways in which the point locations of the Redwoods may be combined with the probability surface itself. The first of these makes use of the "bubble plot" approach used earlier, overlaying the tree locations, whereas the second "drapes" a map of tree locations over a three-dimensional view of the surface. Both of these demonstrate the ability of GIS to combine two data sets from the same region on the same map. In this situation, the auxiliary data was simply the original point data set, but if the data were readily available, it just as easily could have been information about slope or aspect of the ground, soil acidity, or any other factor which might influence the likelihood of a tree surviving in a given location.

Thus, the technique outlined in this paper demonstrates that there are methods in statistics that are "*GISable*," using Openshaw's terminology (Openshaw, 1994). The questions that remain to be answered are mostly in terms of how much further, and in what directions, should the union between GIS and statistics go? For example, would it be possible to extend the cartographic aspects of inspecting the relationship between the density function and other geographical factors to a more rigorous statistical test? Could probability surfaces be used as the basis for simulations allowing GIS to provide "what-if" modeling with a spatial, stochastic element? It is hoped that this paper, and others in the same spirit

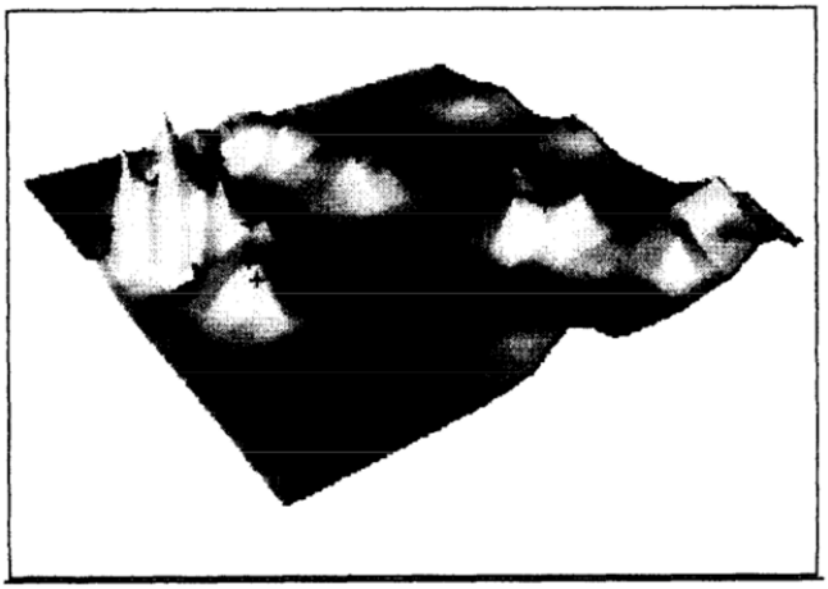

Figure 16. ARC/INFO image of density surface showing Redwood sapling locations.

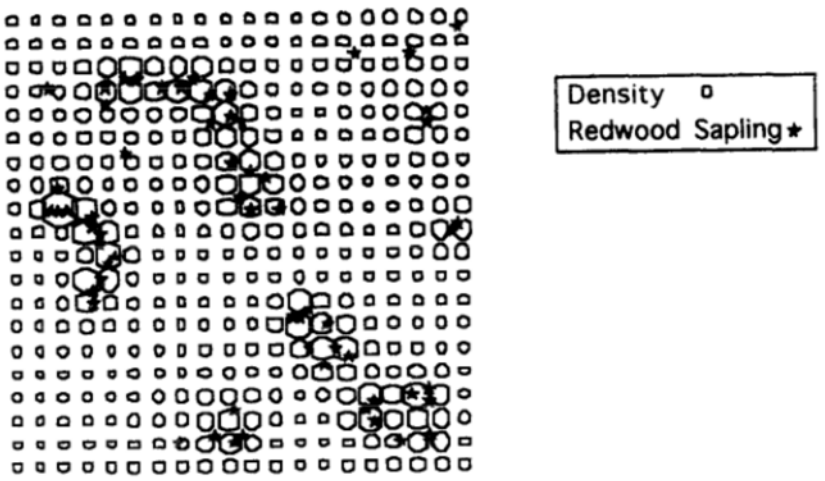

Figure 17. Bubble diagram with Redwood sapling locations from ARC/INFO.

will encourage the publication of more research and practical examples of the benefits of developing links between statistical analysis and the use of GIS.

REFERENCES

Anselin, L., 1988, Spatial econometrics, methods and models: Kluwer Academic Publ., Dordrecht, 284 p.

Baxter, R. S., 1976, Computer and statistical techniques for planners: Methuen, London, 331 p.

Diggle, P. J., 1983, Statistical analysis of spatial point patterns: Academic Press, London, 148 p.

Diggle, P. J., 1985, A kernel method for smoothing point process data: Applied Statistics, v. 34, no. 2, p. 138–147.

Diggle, P. J., 1990, A point process modelling approach to raised incidence of a rare phenomenon in the vicinity of a pre-specified point: Jour. Roy. Statis. Soc., Ser. A, v. 53, no. 3, p. 349–362.

Fotheringham, A. S., and Rogerson, P. A., 1993, GIS and spatial analytical problems: Intern. Jour. GIS, v. 7, no. 1, p. 3–19.

Gatrell, A., 1994, Density estimation and the visualization of point patterns, *in* Hearnshaw, H. M. and Unwin D. J., eds., Visualization in geographical informatic systems: John Wiley & Sons, New York, p. 65–75.

Greig, D. M., 1980, Optimisation: Longman, London, 179 p.

Hooke, R., and Jeeves, T. A., 1961, Direct search solution of numerical and statistical problems: Jour. ACM, v. 8, no. 2, p. 212–229.

Hosking, R. J., Joyce, D. C., and Turner, J. C., 1981, First steps in numerical analysis: Hodder and Stoughton, London, 202 p.

Langford, M., and Unwin, D. J., 1994, Generating and mapping population density surfaces within a Geographical Information System: Cartographic Jour., v. 31, no. 1, p. 21–26.

Openshaw, S., 1990, A spatial analysis research strategy for the regional research laboratory initiative: RRL Initiative Discussion Paper 3, Dept. of Town and Regional Planning, University of Sheffield, 26 p.

Openshaw, S., 1994, A concepts-rich approach to spatial analysis, theory generation, and scientific discovery in GIS using massively parallel computing, *in* Worboys, M. F., ed., Innovations in GIS 1: Taylor & Francis, London, p. 123–137.

The SAS Information Delivery System V 6.09, 1994, The SAS Institute, Cary, North Carolina, 1042 p.

Silverman, B. W., 1986, Density estimation for statistics and data analysis: Chapman and Hall, London, 175 p.

Strauss, D. J., 1975, A model for clustering: Biometrika, v. 62, no. 2, p. 467–475.

APPENDIX

```
/* Listing 1 --- Simple Kernel Density Estimates*/
/* IML Code to carry out simple density estimates */
/* Listings 1-3 run on SAS version 6.08 & later */
/* The IML code requires an input data set of 2D points, */
/* in variables called x and y. */
/* output is a new data set with variables x y and prob */
/* x and y are points on the grid,  prob the probability */
/* estimate  */
/* Read in the data -  here it is Diggle's Redwood Data */


data redwood;
infile 'd:\research\kernel\redwood.dat';
input x y;
format x 6.2 y 6.2;
x = x/10;
y = y/10;
run;
```

```
/* Define the IML Procedure to run the kernel algorithm */

proc iml;
start kernel(k,s);
   /* k = bandwidth s= points on side of square of
evaluation grid */
   /* here,  sample space assumed to be [0,1]x[0,1] */
   /* Rescale if needed */
use redwood;
read all var({'x' 'y'}) into xvec;
a=j(1,nrow(xvec),1);

           /* create the sample points for estimation */
x = (0:s)/s;
y = (0:s)/s;
x = repeat(x`,s+1,1);
y = shape(repeat(y`,1,s+1),(s+1)##2);
grid = x || y;

/* use the tableau method to estimate densities */
d1 = xvec[,1]@j(1,(s+1)##2,1)-(a@grid[,1])`;
d2 = xvec[,2]@j(1,(s+1)##2,1)-(a@grid[,2])`;
d = exp(-(d1##2)/(k##2) -(d2##2)/(k##2))/(k##2)/1.77245381;
/* Guassian Kernel used here */

prob = grid || d[:,]`;       /* Create a new data set with
x,y and densities on grid */
create probs from prob [colname = {x y prob}];
append from prob [colname = {x y prob}];
finish;

run kernel(0.06,50); /An example run */
quit;

/* Use gcontour to draw surface */
proc gcontour;
pattern7 value=solid color=cxff0000;
pattern6 value=solid color=cxcc0000;
pattern5 value=solid color=cxaa0000;
pattern4 value=solid color=cx880000;
pattern3 value=solid color=cx440000;
pattern2 value=solid color=cx220000;
pattern1 value=solid color=cx110000;
plot x*y=prob/pattern join;
run;
```

```
/* Listing 2 --- Cross Validation Likelihood Estimation */
/* Data input requirements are as in listing 1 */


/* Golden Section Search */

proc iml;
start kernel(kmin,kmax);

use redwood;
read all var({'x' 'y'}) into xvec;

a=j(1,nrow(xvec),1);
d1 = xvec[,1]@a-(a@xvec[,1])`;
d2 = xvec[,2]@a-(a@xvec[,2])`;

gs2 = (sqrt(5)-1)/2;
gs1 = 1 - gs2;

k1 = kmin;
k2 = kmin+gs1*(kmax-kmin);
k3 = kmin+gs2*(kmax-kmin);
k4 = kmax;
```

```
run xval(k2,d1,d2,xvec,f2);
run xval(k3,d1,d2,xvec,f3);


/* Main search routine */

do i = 1 to 15;
 if f2 > f3 then do;
   k4 = k3;
   f3 = f2; k3 = k2;
   k2 = k1+gs1*(k4-k1);
   run xval(k2,d1,d2,xvec,f2);
 end;
 else do;
   k1 = k2;
   f2 = f3; k2 = k3;
   k3 = k1+gs2*(k4-k1);
   run xval(k3,d1,d2,xvec,f3);
 end;
end;

/* Choose an optimum */

kopt = (k3+k2)/2;
print 'K Optimum = ' kopt;
finish;



/* Cross-validation likelihood computation for search */

start xval(k,d1,d2,xvec,prob);
 d = exp(-(d1/k)##2 -(d2/k)##2)/(k##2)/1.772453851;
 d = d # (j(nrow(xvec), nrow(xvec)) - i(nrow(xvec)));
 prob = log(d[:,]`);
 prob = prob[+];
finish;


/* Run example program --- Guess at optimal kernel
bandwidth between 0.05 and 0.07 */

run kernel(0.05,0.07);
quit;


/* Listing 3 - IML code for adaptive Kernel estimation  */
/* Data input requirements as in listing 1 */

/* Define the main IML Procedure */
proc iml;
start adrad(k,f,alpha,kmat);
          /* Calculate the adaptive kernel radii */
logf = log(f);
gmean = exp(logf[:]);
kmat = k*(f/gmean)##alpha;
print gmean;
finish;

start adkern(k,s, alpha);
 /* k = bandwidth s= points on side of square of evaluation
grid */
 /* alpha = power correction factor for adaptive method
*/
 /* here,  sample space assumed to be [0,1]x[0,1]
*/
 /* Rescale if needed
*/
use redwood;
```

```
read all var({'x' 'y'}) into xvec;
          /* First,  let create the 'base' estimate */
a=j(1,nrow(xvec),1);
d1 = xvec[,1]@a-(a@xvec[,1])`;
d2 = xvec[,2]@a-(a@xvec[,2])`;
f = exp(-(d1/k)##2 -(d2/k)##2)/(k##2)/1.772453851;
f = f[:,];

run adrad(k,f,alpha,kmat);
            /* Calculate the adaptive radii            */
            /* Create the sample points for estimation */

kmat = repeat(kmat,(s+1)##2)`;

x = (0:s)/s;
y = (0:s)/s;
x = repeat(x`,s+1,1);
y = shape(repeat(y`,1,s+1),(s+1)##2);
grid = x || y;

             /* use the tableau method to estimate
densities */
d1 = xvec[,1]@j(1,(s+1)##2,1)-(a@grid[,1])`;
d2 = xvec[,2]@j(1,(s+1)##2,1)-(a@grid[,2])`;
d = exp(-(d1##2)/(kmat##2)
-(d2##2)/(kmat##2))/(kmat##2)/1.77245381;
prob = grid || d[:,]`;
create probs from prob [colname = {x y prob}];
append from prob [colname = {x y prob}];
finish;
run adkern(0.06,50,-1.5); /* An example run */
quit;

c
c         Listing 4
c
c  Library for kernel estimation
c    Chris Brunsdon June 1994
c
c  Simple kernel algorithm
c
c
c      Subroutine KERN(X,N,RADIUS,XS,NS,LS,CVL)
c
c      X -> Complex array of incidence point locations
c      N -> Dimension of X
c      RADIUS -> Kernel Bandwidth
c      XS -> Sampling set of points at which kernel density
is
c
evaluated
c      NS -> Dimension of XS and LS
c      LS -> Likelihoods To be evaluated at the XS points.
c      CVL -> Cross Validation Likelihood of the model
c
c
       subroutine kern(x,n,radius,xs,ns,ls)
       integer n, ns
       real radius, r, ls(ns), kerfun, f
       complex x(n), xs(ns)
       do i = 1, ns
        ls(i) = 0.0
        do j = 1, n
         r = abs(xs(i)-x(j))
         f = kerfun(r,radius)
         ls(i) = ls(i) + f
        end do
       ls(i) = ls(i)/n
       end do
```

```
      return
      end
c
c     function  CVL(X,LS,N,RADIUS)
c
c     X -> Complex array of incidence point locations
c     LS -> Likelihood at each of the X's
c     N -> Dimension of X and LS
c     RADIUS -> Kernel Bandwidth
c     CVL -> Cross Validation Likelihood of the model
c
c
      function cvl(x,ls,n,radius)
      integer n
      real radius, r, cvl, kerfun, scl, f, ls(n)
      complex x(n)
      cvl = 0.0
      do i = 1, n
       ls(i) = 0.0
       do j = 1, n
        r = abs(x(i)-x(j))
        f = kerfun(r,radius)
        ls(i) = ls(i) + f
        if (i.eq.j) scl = f
       end do
      cvl = cvl+log((ls(i)-scl)/(n - 1))
      ls(i) = ls(i)/n
      end do
      return
      end

c
c
c  Kernel function...change as needed.
c
c
      function kerfun(r,radius)
      real kerfun, r, radius, temp
      temp = r/radius
      if (temp .lt. 3.0) then
       kerfun = exp(-
(r/radius)**2)/(radius**2*1.772453851)
      else
       kerfun = 0.0
      end if
      return
      end
c
c
c     Adaptive kernel algorithm
c
c     subroutine KERNA(X,N,RADIUS,XS,NS,LS)
c
c     X -> Sample grid. Complex array
c     N -> Dimension of X and RADIUS
c     RADIUS -> Bandwidth.  In this case an array also.
c     XS -> Evaluation point complex array
c     NS -> Dimension of XS and LS
c     LS -> The computed likelihoods
c
c
      subroutine kerna(x,n,radius,xs,ns,ls)
      integer n, ns
      real radius(ns),r,ls(ns), kerfun, f
      complex x(n), xs(ns)
      do i = 1, ns
       ls(i) = 0.0
       do j = 1, n
```

```
          r = abs(xs(i)-x(j))
          f = kerfun(r,radius(j))
          ls(i) = ls(i) + f
         end do
        ls(i) = ls(i)/n
        end do
        return
        end
c
c Subroutine to create a sampling grid
c
c      subroutine SGRID(CORNER,SIDE,GRID,M,N,MTN)
c      CORNER -> Complex.  Bottom left corner of grid
c      SIDE -> Side (or lattice length) of grid
c      GRID -> A complex array of dimension MTN.  This will
be the
c
grid.
c      M -> Grid width
c      N -> Grid Height
c      MTN -> M times N !
c
       subroutine sgrid(corner,side,grid,m,n,mtn)
       integer m,n
       complex corner, grid(1:mtn)
       real side
       do i = 1, m
        do j = 1, n
         grid((i-1)*n+j) = corner+side*cmplx(i-1,j-1)
        end do
       end do
       return
       end
c
c
c     subroutine FIXRAD(LS,AR,N,RADIUS,POWER)
c
c     LS -> Likelihood function from a fixed kernel method
c     AR -> Adaptive radius
c     N -> Dimension of LS and AR
c     RADIUS -> Radius from initial fixed kernel estimate
c     POWER -> Exponent on correction term
c
c
      subroutine fixrad(ls,ar,n,radius,power)
      integer n
      real gml, ls(n), ar(n), radius, power
      gml = 0.0
      do i = 1, n
       gml = gml + log(ls(i))
      end do
      gml = exp(gml/n)
      do i = 1, n
       ar(i) = radius*(ls(i)/gml)**power
      end do
      return
      end

c
c      function  ACVL(X,LS,N,RADIUS)
c
c      X -> Complex array of incidence point locations
c      LS -> Likelihood at each of the X's
c      N -> Dimension of X and LS
c      RADIUS -> Kernel Bandwidth array
c      CVL -> Cross Validation Likelihood of the model
c
c
       function acvl(x,ls,n,radius)
```

```
      integer n
      real radius(n), r, acvl, kerfun, scl, f, ls(n)
      complex x(n)
      acvl = 0.0
      do i = 1, n
       ls(i) = 0.0
       do j = 1, n
        r = abs(x(i)-x(j))
        f = kerfun(r,radius(j))
        ls(i) = ls(i) + f
        if (i.eq.j) scl = f
       end do
      acvl = acvl+log((ls(i)-scl)/(n - 1))
      ls(i) = ls(i)/n
      end do
      return
      end


c
c         Listing 5.
c  Adaptive Bandwidth Kernel Density Estimation
c
c       Chris Brunsdon May 1994
c
c
      integer n, dmax
      complex x(1:500), xs(1:500), centre, simplex(4),
step(4)
      complex grid(1:400)
      real east, north, radius, ls(1:500), ar(1:500), cvl,
acvl
      real fcentre, fsimplex(4), f, gl(1:400)
c
c
c  Main section - read data into a complex variable
c
      open (1,file='redwood.dat')
      open (2,file='kernel.dat')
      n = 0
 100  read (1,*,end=110) east,north
       n = n + 1
       x(n) = cmplx(east/10.0,north/10.0)
       xs(n) = x(n)
       go to 100
 110  continue
c
c  Sets up a four-point simplex
c
c
      centre = (0.05, -1.5)
      step(1) = (0.0,0.1)
      step(2) = (0.01,0.0)
      step(3) = (0.0,-0.1)
      step(4) = (-0.01,0.0)
      do i = 1, 4
       simplex(i) = centre + step(i)
       fsimplex(i) = f(simplex(i),x,ls,ar,n)
      end do
      fcentre = f(centre,x,ls,ar,n)
c
c
c  Hooke and Jeeves maximisation method for alpha and
initial
c  bandwidth.
c
      do while (abs(step(1)).gt.1.0e-
5.or.abs(step(2)).gt.1.0e-4)
       fmax = fcentre
```

```
          dmax = 0
          do i = 1, 4
           if (fsimplex(i).gt.fmax) then
            fmax = fsimplex(i)
            dmax = i
           end if
          end do
          if (dmax.eq.0) then
          do i = 1, 4
           step(i) = step(i)/2.0
           simplex(i) = centre + step(i)
           fsimplex(i) = f(simplex(i),x,ls,ar,n)
          end do
         else
          do i = 1, 4
           simplex(i) = simplex(i) + step(dmax)
           fsimplex(i) = f(simplex(i),x,ls,ar,n)
          end do
          centre = centre + step(dmax)
          fcentre = f(centre,x,ls,ar,n)
         end if
        end do
c
c
c  Finally evaluate the density estimates at sampling
points on a
c
grid
c
c
      call sgrid((0.025,0.025),0.05,grid,20,20, 400)
      call kerna(x,n,ar,grid,400,gl)
      do i = 1, 400
       write (2,*) real(grid(i)), imag(grid(i)), gl(i)
      end do
      end
c
c
c  Cross validation likelihood function used in above
program
c
c
      function f(z,x,ls,ar,n)
      real radius, pwr, ls(n), ar(n), cvl, acvl
      complex z, x(n)
       radius = real(z)
       pwr = imag(z)
       xx = cvl(x, ls, n, radius)
       call fixrad(ls, ar, n, radius, pwr)
       f = acvl(x,ls,n,ar)
      return
      end
```